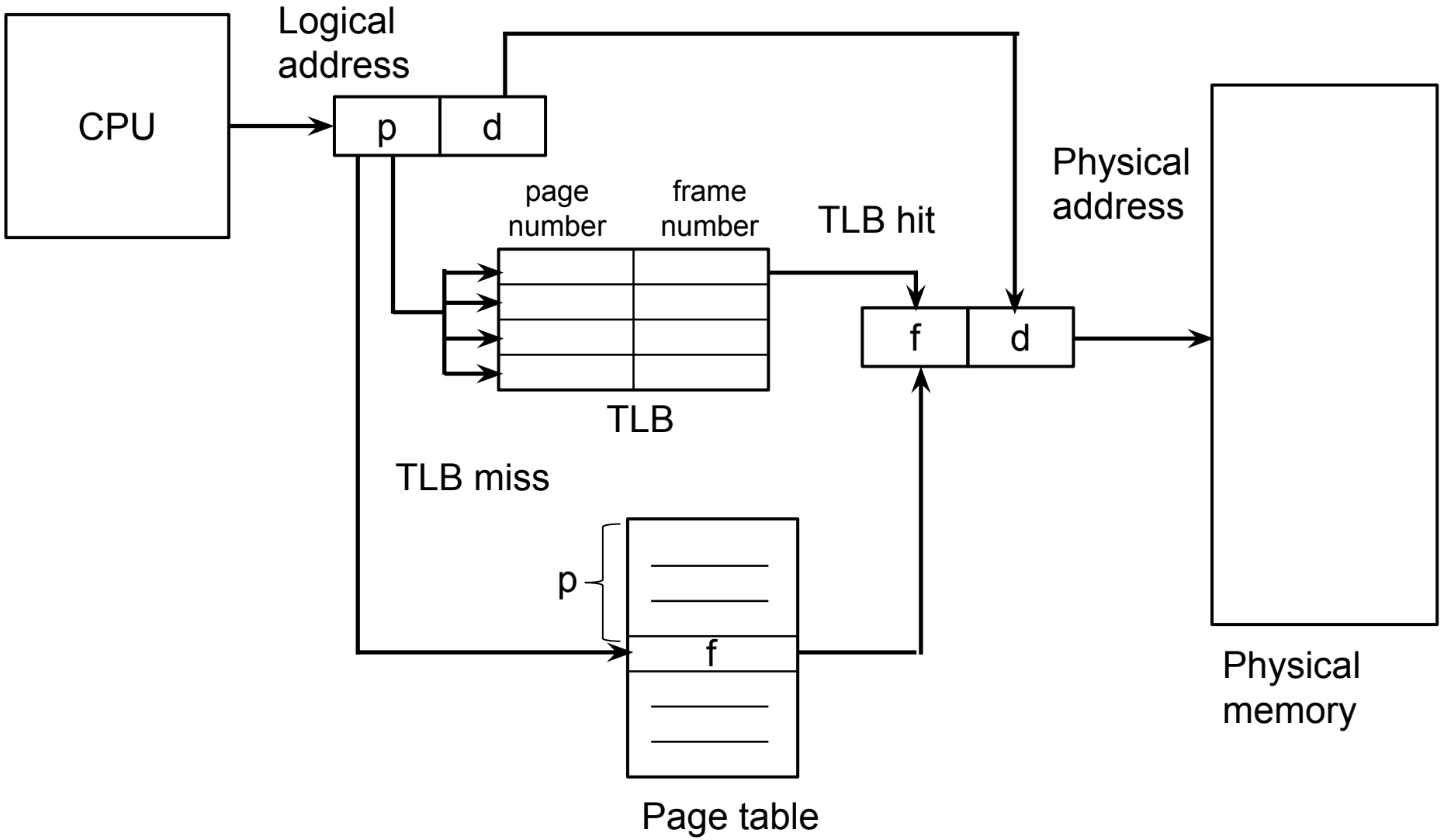
Paging Hardware with TLB



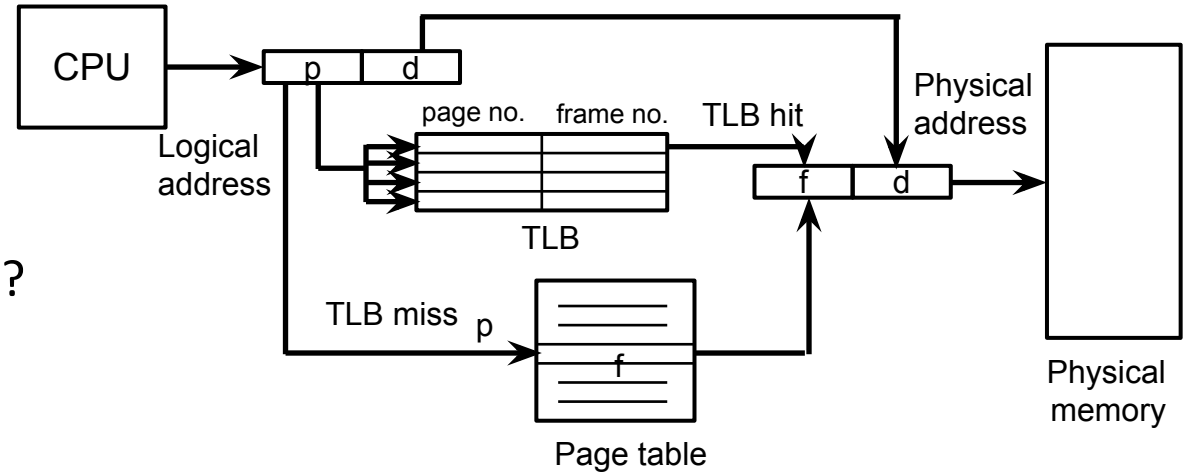
Paging with TLB

- The entire page table was kept in main memory.
- In case of paging the problem is main memory is accessed two times.
- One for finding the frame number in page table and another for accessing the memory address specified by frame number.

- To overcome this problem a high-speed cache is set up for page table entries called a Translation Look aside Buffer (TLB).
- Translation Look aside Buffer (TLB) is nothing but a special cache used to keep track of recently used transactions.
- For a logical address, the processor examines the TLB if a page table entry is present then it is called TLB hit, the frame number is retrieved and then physical address is formed.
- If a page table entry is not found in the TLB then TLB miss will occur, the page number is used to index the page table.

Paging with TLB example

A paging scheme uses a Translation Look aside buffer (TLB).
A TLB access takes 20 ns and a main memory access takes 100 ns.
What is the effective memory access time (in ns) if the TLB hit ratio is 80%?
Solution:
TLB access time = 20 ns (given)
Main memory access time
= 100 ns (given)
Hit ratio = 0.8 (given)
Effective memory access time = ?



Effective memory access time = hit ratio × (TLB access time + memory access time) +

(1- hit ratio) × (TLB access time + 2 × memory access time)

Effective memory access time = 0.8 × (20 + 100) + (1- 0.8) × (20 + 2 × 100) ns
= 0.8 × 120 + 0.2 × 220 ns
= 140 ns

If TLB is not used here then the effective memory access time will be 200 ns.

Page Replacement algorithm

1. FIFO(First in First Out) algorithm
2. Optimal Page replacement algorithm
3. LRU(Least Recently Used) algorithm

Address sequence to reference string generation

The following address sequence is given

0701, 0050, 0120, 0243, 0052, 0322, 0045, 0450, 0266, 0288, 0312, 0042, 0315, 0216, 0125, 0245, 0087, 0068, 0129, 0743, 0055, 0128

Page size = 100 bytes

00 – 99 in Page 0
100 – 199 in Page 1
200 – 299 in Page 2
300 – 399 in Page 3
400 – 499 in Page 4
500 – 599 in Page 5
600 – 699 in Page 6
700 – 799 in Page 7
800 – 899 in Page 8

The reference string is

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

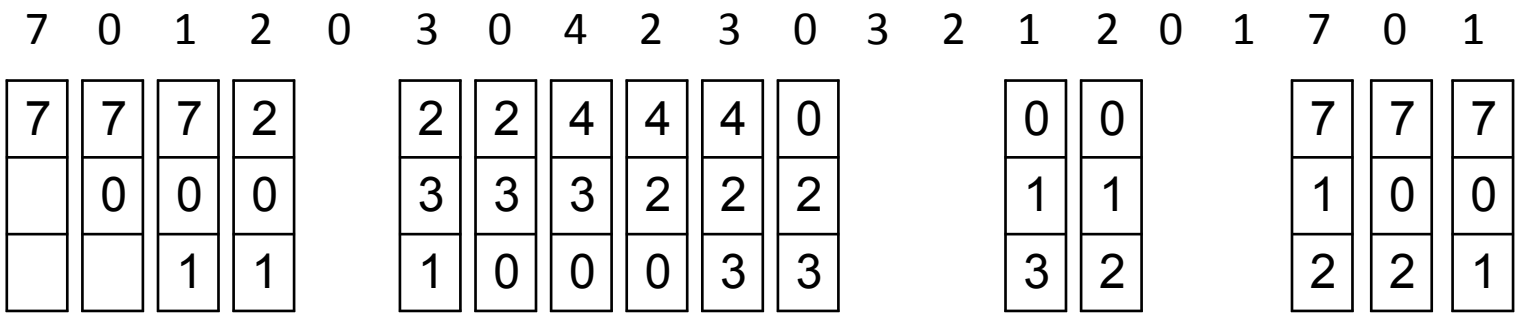
FIFO Page replacement Algorithm

In FIFO page replacement algorithm, which page come first that will be replaced first.

Reference String

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

Available memory frame = 3



Number of page fault = 15

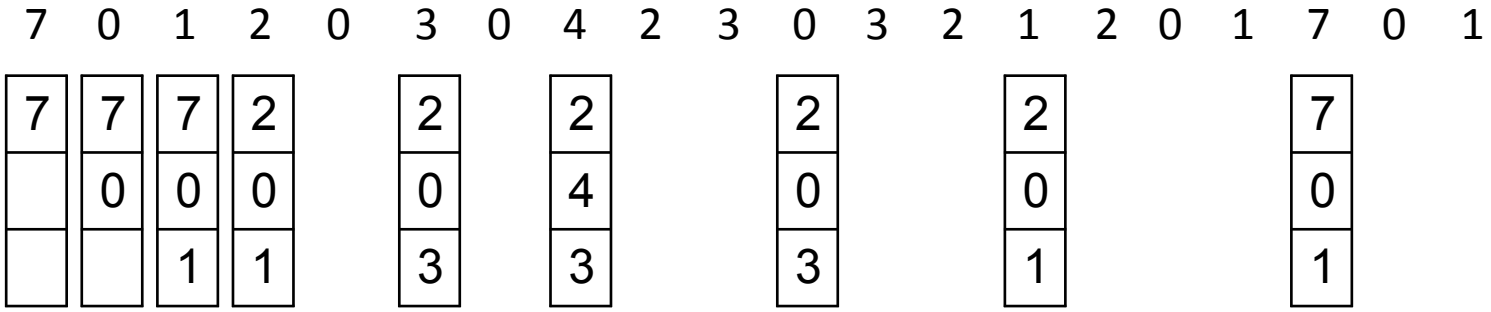
Optimal Page replacement Algorithm

In Optimal page replacement algorithm, replace the page that will not be used for the longest period of time.

Reference String

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

Available memory frame = 3



Number of page fault = 9

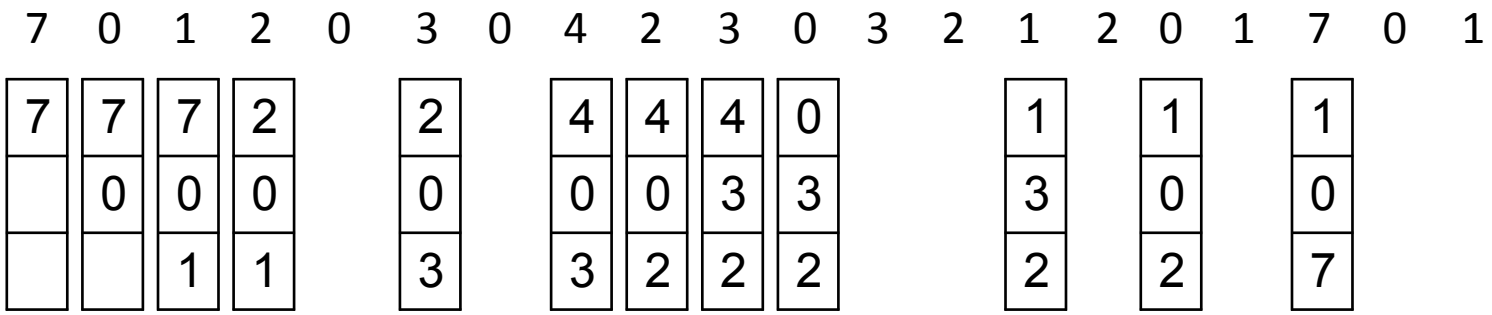
LRU Page replacement Algorithm

In LRU page replacement algorithm, replace the page that has not been used for the longest period of time.

Reference String

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

Available memory frame = 3



Number of page fault = 12

Thank You